P515690.PDF [Page: 3 of 44]

Copy No. 9

DEFENCE R&D DÉFENSE

# RANDI IDL GRAPHICS DEVELOPMENT

*Colin Calnan*

*XWAVE*
*5475 Spring Garden Road*
*Suite 601*
*Halifax, Nova Scotia, Canada*
*B3J 3T2*

**Defence R&D Canada**
Contractor Report
DREA CR 2000-160
December 2000

National Defence / Défense nationale

Canada

'THIS IS AN UNEDITED REPORT ON SCIENTIFIC OR TECHICAL WORK CONTRACTED BY THE DEFENCE RESEARCH ESTABLISHMENT ATLANTIC OF THE RESEARCH AND DEVELOPMENT BRANCH OF THE DEPARTMENT OF NATIONAL DEFENCE, CANADA.

THE CONTENTS OF THE REPORT ARE THE RESPONSIBILITY OF THE CONTRACTOR AND DO NOT NECESSARILY REFLECT THE OFFICIAL POLICIES OF THE DEPARTMENT OF NATIONAL DEFENCE.

National Defence
Research and Development Branch

Défense nationale
Bureau de recherche et développement

DREA CR 2000-160

# RANDI IDL Graphics Development

Colin Calnan

xwave
5475 Spring Garden Road
Suite 601
Halifax, Nova Scotia, Canada
B3J 3T2

Scientific Authority
F. Desharnais

W7707-007913/001/HAL
Contract Number

December 2000

CONTRACTOR REPORT

Prepared for

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canada

## ABSTRACT

The DREA model RANDI (v3.1) produces as output a number of ASCII and binary files, depending on the manner in which its various parts are run. On its original platform, a Sun SPARC Workstation, the model was supported by a suite of FORTRAN / Sun GKS programs that produced graphics output from the model's output data files. Although RANDI has been converted to run on an HP workstation, the graphics programs cannot be as simply converted because of their reliance on the machine-specific Sun GKS library. However, since the target computer has IDL installed, it was decided to rewrite the graphics programs in IDL. The contract that allowed RANDI to be converted to run on the HP also allowed a start to be made on rewriting the graphics programs in IDL. The current contract allowed for an "upgrade" of the existing IDL routines and the writing of some new ones.



## RÉSUMÉ

Selon la façon dont on utilise ses diverses composantes, le modèle RANDI (v3.1) présente ses données de sortie dans un certain nombre de fichiers ASCII et binaires. Sur sa plate-forme d'origine, soit un poste de travail Sun SPARC, le modèle était soutenu par une série de programmes FORTRAN / Sun GKS qui produisaient des sorties graphiques à partir des fichiers de données de sortie du modèle. Le modèle RANDI a été modifié pour tourner sur un poste de travail HP, mais les programmes graphiques ne peuvent pas être modifiés aussi simplement, car ils dépendent de la bibliothèque Sun GKS propre à l'ordinateur. Toutefois, comme le langage de description IDL est installé sur l'ordinateur HP, on a décidé de réécrire les programmes graphiques en IDL. Le contrat de modification de RANDI pour le faire exécuter sur l'ordinateur HP a aussi permis de commencer à réécrire les programmes graphiques en IDL. Le contrat actuel prévoyait une « mise à niveau » des routines IDL existantes et l'écriture de nouvelles routines.



Sun® is a registered trademark of Sun Microsystems, Incorporated
SPARC™ is a trademark of Sun Microsystems, Incorporated
IDL® is a registered trademark of Research Systems, Inc.
Macintosh™ is a trademark licensed to Apple Computer, Incorporated
PostScript® is a registered trademark of Adobe Systems, Incorporated
VMS™ is a trademark of Digital Equipment Corporation
Windows® is a registered trademark of Microsoft Corporation

## EXECUTIVE SUMMARY

RANDI 3.1 is an underwater ambient noise model. It can be used to predict acoustic noise levels and 3-D directionality for shallow and deep water environments, and to predict the low to mid frequency response of sonar receivers to that noise field.

RANDI 3.1 was originally written to run on a Sun SPARC Workstation supported with Sun GKS graphics library. The model had been converted for an HP environment to be used more readily at DREA for sonar research. Under this contract, the RANDI graphics routines were rewritten in IDL to exclude the dependence of the original programs to the Sun GKS library.

As a result, DREA has a functional copy of RANDI ready to be used to model 3-D ambient noise fields for sonar research..

## SOMMAIRE

RANDI 3.1 est un modèle de bruit ambiant sous-marin. Il peut être utilisé pour prédire les niveaux de bruit acoustique et leur directionalité en trois dimensions pour des environnements d'eau profonde et peu profonde, et pour prédire la réponse acoustique de récepteurs sonar à ce champ de bruit aux fréquences basses et moyennes.

Sur sa plate-forme d'origine, soit un poste de travail Sun SPARC, le modèle était soutenu par une bibliothèque de graphiques Sun GKS. Le modèle a été modifié pour tourner sur un poste de travail HP, pour facile à utilisé pour la recherche sonar au CRDA. Sous le contrat actuel, les programmes graphiques ont été réécrit en IDL pour exclure la dépendance des logiciels originels à la bibliothèque Sun GKS.

Le CRDA a maintenant une copie fonctionnelle de RANDI prête à être utilisée pour modeliser les champs de bruit ambiant en trois dimensions pour la recherche sonar.

## Table of Contents

1.0 INTRODUCTION ..... 1
2.0 PRODUCING GRAPHICS ..... 3
2.1 Running **plotRANDI** ..... 3
2.2 General Rules Related to **plotRANDI** ..... 4
3.0 MODIFICATIONS TO INHERITED **plotRANDI.pro** ..... 6
4.0 **plotRANDI** EXPANSION ..... 9
5.0 **plotRANDI** CONCERNS ..... 10
5.1 **plotShipNoise** ..... 10
5.2 **plotAACDF** ..... 10
5.3 **plot3DTLPL** ..... 11
6.0 **plotRANDI** - FORTRAN/GKS CODE COMPARISONS ..... 12
7.0 SUGGESTIONS FOR FURTHER WORK ..... 14
8.0 REFERENCES ..... 16

APPENDIX A - SETUP REQUIREMENTS ..... 17
APPENDIX B - HELP INFORMATION ..... 19

# 1.0 INTRODUCTION

Besides common English typographic conventions, the following conventions are used in this document:

- **bold text** is used for filenames (e.g. **Program.f**)
- **bold Arial text** is used for program names (e.g. **Program**)
- **Arial text** is used for subroutine names (e.g. **Subroutine**)
- ***Arial italics text*** is used for variable names, switches, etc. (e.g. ***SumVar, /H***)
- `Courier text` is used for computer program and data file excerpts (e.g. `IF (X EQ 2)`), text to be typed on the keyboard (e.g. enter `idl`), and information written by programs to the user (e.g. `Press "Q" to Quit`)
- *italics text* is used for computer names (e.g. *fozzie*)
- ***bold italics text*** is used for directories and pathnames (e.g. ***Fozzie:/local/files***)

The Research Ambient Noise Directionality noise model, version 3.1 (RANDI 3.1) consists of a number of separate programs that produce output in the form of ASCII and/or binary files. The particular output file type and format depends on users' instructions to the model, which affects the specific programs run.

This model was originally prepared to run on a Sun SPARC station, and so a series of programs were written for that platform that would produce graphic output from the RANDI output files. These graphics programs were written in FORTRAN making use of Sun's Graphics Kernel System (GKS) graphics library.

On an earlier contract, DREA had RANDI converted to run on an HP workstation and a start was made on converting the graphics programs. Since the HP did not have a GKS library available but did have IDL installed, the new graphics routines were written in IDL. One advantage to this procedure is that if RANDI is ever converted to any other platform with IDL, the graphics routines can be run without modification on the new platform.

The current contract entailed expanding the IDL graphics code to cover more of the original FORTRAN/GKS routines. The original intent of the contract was to have all the graphics routines converted and modifications made to the RANDI code, however the production of new graphics routines took up much more time than was originally anticipated.

There were several reasons for the time overrun in producing graphics code, several of which were:

- The coding and comments in the original FORTRAN/GKS routines were often baffling and occasionally wrong (e.g. code was copied from one routine to another and changed a bit without having the comments changed).
- Instead of simply reading in RANDI's output files and producing plots, the graphics programs frequently did appreciable data manipulation and post-RANDI processing.
- The original IDL routines did not check user-supplied parameters or provide any online "help" information; they were changed to include these missing features.

P515690.PDF [Page: 10 of 44]



The IDL graphics are produced by a number of individual programs, all of which are contained in the file **plotRANDI.pro**. For that reason, this report refers to the graphics code as "the program **plotRANDI**" even though, technically speaking, it isn't. This structure also mimics the model RANDI, which itself consists of a number of individual programs but which together are considered as one model. The original IDL code was produced as one file in this manner (which does have both advantages and disadvantages), and since there was no overwhelming reason to split the code into separate routines this structure was maintained. In later sections the individual programs that comprise **plotRANDI** are referred to by name, e.g. **plotBeamNoise** and **plot3DTLPL**.

The original IDL routine **plotRANDI** (the filename **plotRANDI.pro** was retained) consisted of fewer than 1000 lines, but the version that existed at the end of this contract was almost 5000 lines long with a separate "help" information file of about 550 lines. Sections 3 and 4 of this report contain more information on the original IDL program and changes and additions that were made to it.

This report is a description of how to use **plotRANDI** to produce graphics, changes and expansion made to the original **plotRANDI**, concerns with the program as it exists, comparison with the original FORTRAN/GKS graphics routines, and suggestions for further work. As well, two appendices describe how to get IDL working on various platforms and present the file used for help information. The help file **plotRANDI.help** is designed to be, in effect, a user's guide to using the program as well as specifying the parameters used by the various graphics programs; it is presented in Appendix B.

The majority of the information in this report concerning RANDI came from *RANDI 3.1 User's Guide* [Breeding et al, 1994]. As well, determination of data file formats came from the *User's Guide* and from examining the data files themselves and the programs that created them and read them. The routines of **plotRANDI** were based on the FORTRAN/GKS programs that were originally used to produce the graphics on the SPARC workstations in order to mimic their output.

# 2.0 PRODUCING GRAPHICS

## 2.1 Running plotRANDI

To run the program either:
- the user must be in the directory with **plotRANDI.pro** and **plotRANDI.help**, or
- the directory holding those files must be in the accounts path.

See Appendix A for information on the above.

Assuming one of the above conditions is met, to plot data the user must:
- Enter the command "`idl`" at the command line prompt to start IDL.
- Enter "`.com plotRANDI`" at the IDL prompt to compile all the routines that comprise **plotRANDI**; the routines will be listed to the screen as they are compiled. This step is crucial.
- Enter the name of the routine that produces the desired graphic.

Table 2.1, below, lists the routines that produce graphics from RANDI output, a description of the output plots, and the page in the *RANDI 3.1 User's Guide* that has a sample of the output plot.

| Program | Output | Sample Page |
|---|---|---|
| **plotBeamNoise** | Polar plot of beamformed noise | 69 |
| **plotShipNoise** | Polar plot of spiked or sinc smoothed noise level due to individual ships | 70, 71 |
| **plotHzDir** | Polar plot of horizontal directionality averaged over all dead reckon increments | None |
| **PlotTLSL** | Transmission loss or signal level vs. range for one dead reckon | 67 |
| **plotDir3D** | 3-D plot of noise level over azimuth vs. elevation angle, and line plots of vertical and horizontal directionality | 74 |
| **plotAACDF** | Percentage occurrence in azimuth of spatial noise holes as a function of beamwidth | 75 |
| **plotAmNoise** | 3-D plot of shipping and wind noise over azimuth and depth, up to 3 polar plots of horizontal directionality at specified depths, and a plot of noise vs. depth | 73 |
| **plotOmniSN** | Plot of coherent or incoherent omnilevel vs. hydrophone | 72 |
| **plot3DTLPL** | 3-D plot of transmission loss or pressure level by depth and range | 68 |

**Table 2.1: RANDI Graphics Programs**

Appendix B contains a listing of the file **plotRANDI.help**, which contains usage information on all the above programs.

If the user wishes to produce a number of plots and knows what command line parameters are required, then an IDL routine can be written to create the required plots. For example, if a user wishes to make PostScript plots of **plotOmniSN** output for all six dead reckons and for both coherent and incoherent data from the input file **scotia.in**, then a local IDL file named, for example, **do_it.pro** can easily be created that consists of:

```
pro do_it

plotomnisn, "scotia.in", dr=1, ctyp=1, ps="plot11.ps"
plotomnisn, "scotia.in", dr=1, ctyp=2, ps="plot12.ps"
plotomnisn, "scotia.in", dr=2, ctyp=1, ps="plot21.ps"
plotomnisn, "scotia.in", dr=2, ctyp=2, ps="plot22.ps"
plotomnisn, "scotia.in", dr=3, ctyp=1, ps="plot31.ps"
plotomnisn, "scotia.in", dr=3, ctyp=2, ps="plot32.ps"
plotomnisn, "scotia.in", dr=4, ctyp=1, ps="plot41.ps"
plotomnisn, "scotia.in", dr=4, ctyp=2, ps="plot42.ps"
plotomnisn, "scotia.in", dr=5, ctyp=1, ps="plot51.ps"
plotomnisn, "scotia.in", dr=5, ctyp=2, ps="plot52.ps"
plotomnisn, "scotia.in", dr=6, ctyp=1, ps="plot61.ps"
plotomnisn, "scotia.in", dr=6, ctyp=2, ps="plot62.ps"

end
```

Once this file is created, it can be compiled in IDL via the command "`.com do_it`" and then run by entering "`do_it`". By omitting the "`ps`" option screen graphics can be produced, but the screens will normally flash by too quickly to be examined.

## 2.2 General Rules Related to plotRANDI

A number of general rules were decided upon in setting up the method by which parameters are passed to the graphics programs. As well, IDL imposes some restrictions on the running of programs. The following list combines the two sets of rules.

- IDL is not case sensitive with regard to commands or switches but it is with regard to filenames on case sensitive operating systems. Therefore when compiling an IDL file the filename must be entered exactly as it appears in a file listing. However, since IDL itself is not internally case sensitive, a file named **plotDooDah.pro** containing the program **plotDooDah** must be compiled via the command "`.com plotDooDah`", but can be run by issuing the commands "`plotdoodah`", "`PLOTDOODAH`", "`PlOtDoOdAh`", etc. This means that the programs listed in Table 2.1 need not be entered with the cases as shown. As can be imagined, this feature leads itself to great possibilities of confusion for a case-unwary programmer.
- All mandatory parameters are entered without keywords
- When a program has more than one mandatory parameter, they must be entered in the order specified.

- Many programs use the parameter *DR*, which is the number of a dead reckon whose data are to be plotted. Since the number of dead reckons in the data can vary from run to run, entering a *DR* value of 0 or less causes the program to tell the user how many dead reckons worth of data there are, after which the user must select a value.
- Many programs allow the entry of a minimum and maximum value for an axis for the plotting of data (by default the axis parameters are set to include all the data). Most of those routines allow the user to enter the same value for both, after which the routines present the user with the minimum and maximum data values and require that axis minimum and maximum values be entered.
- All 3-D plotting routines produce colour output by default. However they all have the switch */BW*, which by its presence causes black and white graphics to be produced instead.
- All programs have the optional parameter *PS* that allows the specification of an output PostScript filename. Using this parameter means that the output graphics will be written to the named file instead of being drawn to the screen. If the filename provided is that of a file that already exists, the existing file will be overwritten.
- All programs have the switch */H*, which overrides all other switches, causes the help information for that program to be written out, and ends the program.

## 3.0 MODIFICATIONS TO INHERITED plotRANDI.pro

This section specifies changes that were made to the file **plotRANDI.pro** as an earlier contractor left it. Several of the reasons for making changes were:

- simplifying operations for the user;
- "good programming" changes (e.g. adding error checking and help information, structuring the code);
- producing plots that more closely matched those in *RANDI 3.1 User's Guide*;
- performing calculations that existed in the original FORTRAN/GKS routines but were missing from the original version of **plotRANDI.pro**; and
- correcting bugs in the original.

At the beginning of this contract **plotRANDI.pro** contained the following:

| | |
|---|---|
| *mainInputStruct* | - (data structure) contains data read from the main RANDI input file |
| *omniStruct* | - (data structure) holds omni level data as read in the function loadMainOutput |
| loadMainInput | - (function) reads data from the main RANDI input file |
| loadMainOutput | - (function) reads data from the main RANDI output file |
| loadVertDir | - (function) reads RANDI's vertical directionality output file |
| drawCompass | - (procedure) draws some polar plot common background material |
| writeRunInfo | - (procedure) writes common data to several plots in a consistent format |
| initPlot | - (function) initializes graphics output |
| closePlot | - (procedure) closes a PostScript output file (if used) and restores graphics to pre-initPlot conditions |
| **plotBeamNoise** | - (program) PLOT BEAM NOISE |
| **plotShipNoise** | - (program) PLOT SHIP NOISE |
| **plotHzDir** | - (program) PLOT HoriZontal DIRectionality |
| **plotTLSL** | - (program) PLOT Transmission Loss/Signal Level |
| **plotDir3D** | - (program) PLOT ambient noise DIRectionality (3-D) |

The following table lists the above routines and any changes that were made to them during the course of this contract. All changes are documented in the code.

| Routine | Changes |
|---|---|
| *mainInputStruct* | • Added two items needed by new plotting routines. |
| *omniStruct* | None. |
| loadMainInput | • Added error checking on the file opening<br>• Made it read two more parameters<br>• Changed it from a procedure to a function, thus allowing the return of an error code |

| Routine | Changes |
|---|---|
| loadMainOutput | • Added error checking on the file opening<br>• If *DR* is passed in less than 1, made the routine read through the file to count the dead reckons, inform the user of how many there are, and require that the user select one<br>• If there are no dead reckons the routine writes out a message and returns with an error flag<br>• Changed the reading looping structure from `GOTO` to a `while` loop<br>• Separated invalid parameter list and invalid (i.e. too high) *DR* selection error messages |
| loadVertDir | • Added error checking on the file opening<br>• Added items to the parameter list to pass in more complete instructions and to allow the return of more data<br>• The routine originally assumed that all beams would be plotted; code was added to set up the heading values based on other possibilities |
| drawCompass | None |
| writeRunInfo | • Added items to the parameter list to pass in more complete instructions<br>• Added a label indicating the beamform data source, if that's what is being plotted<br>• Corrected data unit labels for ship noise and horizontal directionality plots |
| initPlot | • If PostScript output files are being created, allowed plots to be created with 8 bit colour<br>• On screen plots, changed the colour scheme from white lines and text on a black background to black on white |
| closePlot | None |
| **plotBeamNoise** | • Changed the input parameter scheme somewhat, simplifying it<br>• Add an input parameter that causes help information to be written to the screen<br>• Added checking on most of the input parameters<br>• If loadMainInput or loadMainOutput returns with an error flag, the program exits<br>• Added error checking on a file opening<br>• Made the routine write out a message and exit if Volumetric DFT data are in the data files; this was done because the original FORTRAN/GKS routines lacked code on how to plot these data<br>• Based on data read in, the user may be prompted for more specific plot information<br>• Arrays are not now assumed to be non-vertical arrays; different code is executed for each type<br>• Plot min and max values are determined differently<br>• Plot labelling was changed to match that of the original FORTRAN/GKS routine |

P515690.PDF [Page: 16 of 44]



| Routine | Changes |
|---|---|
| plotShipNoise | • Changed the input parameter scheme somewhat, simplifying it<br>• Add an input parameter that causes help information to be written to the screen<br>• Added checking on most of the input parameters<br>• If loadMainInput or loadMainOutput returns with an error flag, the program exits<br>• Added error checking on a file opening<br>• Plot min and max values are determined differently |
| plotHzDir | • Changed the input parameter scheme somewhat, simplifying it<br>• Add an input parameter that causes help information to be written to the screen<br>• Added checking on most of the input parameters<br>• If loadMainInput or loadMainOutput returns with an error flag, the program exits<br>• Plot min and max values are determined differently<br>• Plot labelling was changed to match that of the original FORTRAN/GKS routine |
| plotTLSL | • Renamed to this from tlPlot to match the naming convention of the other routines<br>• Added parameters to be consistent with the FORTRAN/GKS routine<br>• Add an input parameter that causes help information to be written to the screen<br>• Added checking on most of the input parameters<br>• If loadMainInput returns with an error flag, the program exits<br>• Added error checking on a file opening<br>• Corrected a bug and changed range values to km from m<br>• Plot min and max values are determined differently |
| plotDir3D | This was only in rudimentary form, apparently the programmer was just getting started on it when the last contract ended. The routine was expanded to completion, going from 40 to 370 lines in length. |

**Table 3.1: Changes to original plotRANDI routines**

As well, numerous comments were added to the above routines and a generalized header was added to the overall file itself.

## 4.0 plotRANDI EXPANSION

The following list contains information on routines that were added to **plotRANDI** during the course of this contract.

| | |
|---|---|
| **Draw3DLegend** | - (function) draws 3-D plots' legend bars with labels; it may be either in colour or black and white |
| **ShowHelp** | - (function) copies information for a specified plotting routine from **plotRANDI.help** to the screen |
| **GetDBMinMax** | - (function) gets dB min/max values for a plot based on users requests |
| **GetMinMax** | - (function) gets non-dB min/max values for a plot based on users requests |
| **Chk_Par** | - (function) checks the validity of user-provided input parameters |
| **Get_Omni** | - (function) gets an omni level for a specified depth |
| **Deg2str** | - (function) produces a latitude/longitude/angle character string from an input angle in degrees |
| **plotAACDF** | - (program) PLOT the Azimuthal Anisotropic Cumulative Distribution Function |
| **plotAmNoise** | - (program) PLOT Ambient NOISE azimuth vs. depth data |
| **plotOmniSN** | - (program) PLOT OMNI Shipping Noise level |
| **plot3DTLPL** | - (program) PLOT 3-D Transmission Loss/Pressure Level |

P515690.PDF [Page: 18 of 44]



# 5.0 plotRANDI CONCERNS

There are three routines in **plotRANDI** that are known to be incomplete or potentially confusing in one way or another. This section points out these routines and expands on the concerns associated with these routines, which are:

**plotShipNoise** - lacks an option that its FORTRAN/GKS counterpart has
**plotAACDF** - contains commented-out code
**plot3DTLPL** - may be reading some data incorrectly

## 5.1 plotShipNoise

The program does not plot spiked ship data as a smoothed curve, even though the user is allowed to ask for that option and the FORTRAN/GKS version of this program does make that plot. (Actually if the user asks **plotShipNoise** for this plot, the routine returns a message stating that it is not yet produced and then exits.) This option was left out of the routine for two main reasons:

1. The code for producing the data to be plotted from the data as read in is fairly confusing and might take several days to understand and convert.
2. The scientific authority on the project preferred to skip it for now since the other two plot types produced by the routine fulfil all the usual requirements for shipping noise plots.

The FORTRAN/GKS program that contains the code for this routine is **mainplot.f**, and the code for performing the data manipulation required for the missing plot is in lines 622 through 758. A programmer might be assisted by code in routine **plotAmNoise**, where a similar data smoothing takes place. The **plotAmNoise** code was translated from the FORTRAN/GKS routine **dirplt.f**. If **plotShipNoise**'s deficiency is fixed, comments in the routine and in **plotRANDI.help** should be edited to reflect the altered code.

## 5.2 plotAACDF

When this routine was being written its general format was copied from an existing routine. Because of that, two input parameters relating to selecting the Y axis minimum and maximum were added to the command line. They were being written into the code when it was noticed that they did not exist in the FORTRAN/GKS version of the program. Therefore all code relating to these parameters was commented out in **plotRANDI.pro** and **plotRANDI.help**, but was not cut from these files. The unnecessary code was left in place because it was realized that if in the future these parameters and their functionality were ever needed, most of the work required for their addition to the code would be already in place. Comments in the code explain all of this to any programmers who might examine it; users would be totally unaware of the commented code and of this concern.

## 5.3 plot3DTLPL

The FORTRAN/GKS basis for this program **fldplt.f** reads data from, among other files, **fepe.out** and **sspe.out**. From these files it reads the source and receiver latitude and longitude as *slat, slon, slat, slon*, and then writes out *slat, slon* for both the source and receiver latitude and longitude. This means that only the second latitude/longitude pair is saved (the first being overwritten) and that the same position is written out for both source and receiver.

This did not seem right, so programs that use the files **fepe.out** and **sspe.out** were looked for. Three routines were found that write to these files. The programs and their writing instructions are:

| | |
|---|---|
| **pe.f** | - writes *rlat, rlon, slat, slon* into file **sspe.out** |
| **fepe_rad.f** | - writes *slat, slon, shiplat, shiplon* into file **fepe.out** |
| **fepe_ship.f** | - writes *slat, slon, rlat, rlon* into file **fepe.out** |

The scientific authority stated that *shiplat* and *shiplon* in the second case would probably correspond to receiver latitude and longitude, so the implication is that:

| | |
|---|---|
| **sspe.out** | - has receiver latitude/longitude followed by source latitude/longitude |
| **fepe.out** | - has source latitude/longitude followed by receiver latitude/longitude |

However, in writing this routine and checking the data read from the various files, it was discovered that the first latitude/longitude pair read from the file **fepe.out** was identical to the receiver location as specified in the RANDI input file. If this is consistent across all RANDI input / **fepe.out** pairs, it means that **fepe.out** has the receiver position before the source location, in the same manner **sspe.out** seems to have.

This was taken as a basic assumption, so the routine was written to read receiver position first, followed by the source position. Should this assumption be incorrect, it would be easy to change the code to something appropriate, even if the differences are file type specific. To locate the code that reads the positions, a programmer must look for the string "`RLAT, RLON`", which occurs twice in this routine.

If the code does get changed because of later discoveries on this topic, the programmer making the changes is asked to update the comments in **plot3DTLPL.pro** to reflect the new coding. Those comments are essentially the text that appears in this section, from which it was obtained.

The scientific authority has noted that the first time that a user runs RANDI and then plots the results with **plot3DTLPL**, it would become obvious if the program is in error. That is because the user, having set up the input data, would know the source and receiver positions and could easily tell if those written on the plot are correct.

# 6.0 plotRANDI - FORTRAN/GKS CODE COMPARISONS

Table 6.1, below, is a modified subset of Table 7.1 from pages 50 and 51 of *RANDI 3.1 User's Guide*. It matches the IDL routines with their FORTRAN/GKS originals and gives information on the output plots. The table contains:

- **IDL Routine** - the names of the IDL routines in **plotRANDI**; where no IDL routine yet exists as a match for the next column's program, this column contains the entry "None"
- **FORTRAN/GKS** - the names of the original Sun SPARC station programs on which the first column IDL routines are based
- **Plot Title** - a sample plot title on the graphics produced by the routines in columns 1 and 2
- **Description** - a description of the previous column's plot
- **Sample Page** - the page of a sample plot in *RANDI 3.1 User's Guide*; if the manual has no sample this column contains the entry "None"

| IDL Routine | FORTRAN/ GKS | Plot Title | Description | Sample Page |
|---|---|---|---|---|
| plotBeamNoise | mainplot.f | "Beam Noise" | Polar plot of beamformed noise | 69 |
| plotShipNoise | mainplot.f | "Shipping Noise Field" or "Smoothed Shipping Noise Field" | Polar spike plot of noise level due to individual ships & Sinc smoothed polar spike plot of noise level due to individual ships | 70, 71 |
| plotHzDir | mainplot.f | "Time Avg Horizontal Directionality" | Polar plot of horizontal directionality averaged over all dead reckon increments | None |
| plotTLSL | tlplt.f | "FEPE Transmission Loss", "SSPE Transmission Loss", or "FEPE Signal Level" | Transmission loss or signal loss vs. range for the specified radial | 67 |
| plotDir3D | vertfld.f | "Ambient Noise 3-D Directionality" | 3-D plot of noise level over azimuth vs. elevation angle (vertical noise field), and line plots of vertical and horizontal directionality | 74 |
| plotAACDF | aacdf.f | "Azimuthal Anisotropic Cumulative Distribution Function" | Percentage of occurrence in azimuth of spatial noise holes as a function of beamwidth | 75 |

| IDL Routine | FORTRAN/ GKS | Plot Title | Description | Sample Page |
|---|---|---|---|---|
| plotAmNoise | dirpit.f | "Ambient Noise: Azimuth vs. Depth" | 3-D plot of shipping and wind noise over azimuth and depth, up to three horizontal directionality noise plots at selected depths, and a plot of noise vs. depth | 73 |
| plotOmniSN | omni_ship.f | "Hydrophone Shipping Noise Level" | Plots incoherent or coherent omnilevel vs. hydrophone | 72 |
| plot3DTLPL | fldpit.f | "Transmission Loss" or "Pressure Level" | 3-D plot of transmission loss or pressure level over depth and range | 68 |

Table: 6.1: plotRANDI – FORTRAN/GKS Code Comparisons

# 7.0 SUGGESTIONS FOR FURTHER WORK

The first item on a list of further RANDI-related work to be done would be the translation of the remaining FORTRAN/GKS graphics routines to IDL. Table 7.1, below, is a modified subset of Table 7.1 from pages 50 and 51 of *RANDI 3.1 User's Guide*, and contains information on the FORTRAN/GKS graphics routines that have not yet been translated. The table contains:

- **FORTRAN/GKS** - the names of the original Sun SPARC station programs on which the first column IDL routines are based
- **Plot Title** - a sample plot title on the graphics produced by the routines in columns 1 and 2
- **Description** - a description of the previous column's plot
- **Sample Page** - the page number in *RANDI 3.1 User's Guide* of a sample plot

| FORTRAN/ GKS | Plot Title | Description | Sample Page |
|---|---|---|---|
| bathyplt.f | "Geographic View of Bathymetry" | 3-D plot of bathymetry | 61 |
| bathygcp.f | "Geographic View of Bathymetry (and Shipping)" | 3-D plot of bathymetry; the user may choose to plot the receiver location and the extracted and processed ships | 62 |
| hitsplt.f | "Geographic View of Discrete Shipping" | Plot of discrete shipping on a latitude/longitude grid | 63 |
| shipden.f | "Geographic View of Shipping Lanes" | 3-D plot of HITS shipping densities over a latitude/longitude grid; user may also plot discrete ships | 64 |
| ahs.f | "Array Heading Surface" | 3-D plot of SNR gain over broadside for different bearings as a function of array heading; includes supplemental plot of noise rose and optional plot of array heading rose | 65 |
| envplot.f | Environmental Cross-Section for Specific Radial | Plots bathymetry and sound speed profile for a ship or radial; marks critical depth and sound channel axis. | 66 |
| vol_beam_surf.f | Volumetric Array Beam Surface | 3-D plot of beam noise over elevation angle and azimuth | 77 |

**Table: 7.1: Untranslated Fortran/GKS Routines**

A second task could be to resolve the concerns of Section 5.0, expanding **plotShipNoise** to include the missing plot and resolving the source/receiver location confusion in **plot3DTLPL**.

A third task could be to convert the RANDI model to ANSI standard FORTRAN. This would potentially allow the model to be compiled by every FORTRAN compiler at DREA, and so would allow the model to be run on all platforms for which there is a FORTRAN

compiler. Combined with the widespread use of IDL, this would allow the model to be run and graphics produced on the vast majority of the computers at DREA.

P515690.PDF [Page: 24 of 44]



# 8.0 REFERENCES

Breeding, J.E., Pflug, L.A., Bradley, M., Hebert, M., and Wooten, M. (1994). *RANDI 3.1 User's Guide*, Naval Research Laboratory, Report NRL/MR/7176--94-7552, August 19, 1994.

Also used as reference were some of the FORTRAN routines that comprise the model RANDI and those that produced graphics output.

The original **plotRANDI.pro** program and notes made by the previous contractor, Alexander MacGillivary, were extensively used.

# APPENDIX A - SETUP REQUIREMENTS

Two files comprise **plotRANDI**: **plotRANDI.pro**, which contains all the IDL code for the program, and **plotRANDI.help**, a file with all the help information read by the program. These files are normally located on the HP workstation *fozzie* in the directory:

***/local/models/RANDI/randi/disk1/randi3.1/plotRANDI_2000***

To run **plotRANDI** either:

- the files must be in the directory from which you are going to run **plotRANDI**, or
- the directory that the files are in must be in the IDL installation's path.

The files may be placed in a directory that is already in IDL's path but it is recommended that their home directory be added to the path or a new directory be created for them, both of which would simplify the files' upgrade or removal.

The IDL installation's current path is stored in the environment variable *!PATH*, whose value may be found by entering the command "`print, !PATH`" while in an IDL window.

The value of *!PATH* is set in different ways on different platforms, but briefly:

| | | |
|---|---|---|
| UNIX | - | *!PATH* is initialised from the environment variable *IDL_PATH* when IDL starts; this variable may be assigned in the files **.cshrc** or **.profile**. |
| VMS | - | *!PATH* is initialised from the logical name *IDL_PATH* when IDL starts; this variable may be assigned in the file **LOGIN.COM**. |
| Windows | - | *!PATH* is initialised from the saved *IDL for Windows* preferences data or from the DOS environment variable *IDL_PATH* when IDL starts. |
| Macintosh | - | *!PATH* is initialised from the saved *IDL for Macintosh* preferences data when IDL starts. |

In any event, the value of *!PATH* can be changed while in IDL by the simple expedient of an assignment statement, for example, on a UNIX system a user may enter:

```
!PATH=!PATH+' :/local/user/idl-saplot'
```

More detail is given in greater clarity in the IDL documentation. In the V5.1 manuals the information is found in *Reference Guide (Vol 1: A-M)*, Chapter 3, pp. 19-21, under the "IDL Environment System Variables" subsection "!PATH."

As well, the IDL setup may have to be run. That is accomplished on *fozzie* by having the following line in the **.login** file (or whatever the appropriate shell startup file is named):

```
source /local/rsi/idl_5.1/bin/idl_setup
```

If **plotRANDI** is being run from a remote computer, to get the graphics to appear on the screen locally the user must obtain the IP address of the local computer (111.222.33.44, for example) and enter the following command on the remote computer:

```
setenv DISPLAY 111.222.33.44:0.0
```

Note that **plotRANDI** has only been tested on IDL version 5.1, its development version.

# APPENDIX B - HELP INFORMATION

The following is a listing of the **plotRANDI** help file **plotRANDI.help**. Portions of this file are written to the screen each time the user enters a **plotRANDI** routine name with the /H switch. This file is included in this document because it contains information on how to use the various routines that comprise **plotRANDI** and so is a very handy reference.

```
;*****************************************************************************
;*                                                                           *
;*  plotRANDI.help                                                           *
;*                                                                           *
;*  This is the "Help" file for "plotRANDI", and it contains help information *
;*  on all the procedures of that program that are used directly by that     *
;*  program and that have the "/H" option.  This file is read and displayed  *
;*  by the function "ShowHelp".                                              *
;*                                                                           *
;*  This file has help information for the following routines:                *
;*    plotBeamNoise     - PLOT BEAM NOISE                                    *
;*    plotShipNoise     - PLOT SHIP NOISE                                    *
;*    plotHzDir         - PLOT HoriZontal DIRectionality                     *
;*    plotTLSL          - PLOT Transmission Loss/Signal Level                *
;*    plotDir3D         - PLOT ambient noise DIRectionality (3-D)            *
;*    plotAACDF         - PLOT Azimuthal Anisotropic Cumulative Distribution *
;*                        Function                                           *
;*    plotAmNoise       - PLOT AMbient NOISE azimuth vs. depth data          *
;*    plotOmniSN        - PLOT OMNI Shipping Noise level                     *
;*    plot3DTLPL        - PLOT 3-D Transmission Loss/Pressure Level          *
;*                                                                           *
;*  This file communicates information to "ShowHelp" via in-line rules and   *
;*  commands, all of which start in column one of this file.  These rules and *
;*  commands are case sensitive and are:                                     *
;*   ;                  - the line is a comment                             *
;*   !NEWHELP <routine> - the next line starts the help information for the *
;*                        routine "<routine>"; the name is case sensitive    *
;*   !ENDHELP           - the previous line is the last line of help        *
;*                        information for the current routine                *
;*                                                                           *
;*  File designed and started:   2000-06-22   Colin Calnan   xwave solutions *
;*                                                                           *
;*****************************************************************************
;*                                                                           *
;*  Note to me: I should create a mechanism in here and in "ShowHelp" for    *
;*  commonly used groups of lines of text (e.g. for "psfile" and "/H").  This *
;*  would simplify things a bit and make it easier to update descriptions and *
;*  include new items in a number of HELP areas.                             *
;*                                                                           *
;*  This mechanism could take the form of:                                   *
;*   !INCLUDEDEF <include name> - to flag the start of an "include" term's   *
;*                                definition section                         *
;*   !ENDINCDEF                - to flag the end of an "include" term's      *
```

```
;*                                  definition section                          *
;*                                                                              *
;*  The lines between these flags would contain the text associated with the    *
;*  "include" term named "<include name>".  I'd also need:                      *
;*   !INCLUDE <include name>     - whenever this is located in the file, it     *
;*                                 would be replaced in the screen text by the  *
;*                                 text between the !INCLUDEDEF and !ENDINCDEF  *
;*                                 flags for the specified "include" term.      *
;*                                                                              *
;*  "Include" definitions would have to occur at the top of this file, and      *
;*  the appearance of the first !NEWHELP flag would signify that no more        *
;*  "include" definitions will be forthcoming.  If any are found after the     *
;*  first !NEWHELP, they will be treated as part of the help information and    *
;*  written out.                                                                *
;*                                                                              *
;******************************************************************************
;
;  "plotBeamNoise"

!NEWHELP plotBeamNoise

  Note: In the "where:" section below, "default" describes the program's action
        or gives the variable's value as set by the program when the variable
        is not provided by the user.

  plotBeamNoise                                                - PLOT BEAM NOISE

  This routine plots beam noise from the main RANDI output file.

  Warning:  If the beamforming technique used by RANDI was the Volumetric
            Discrete Fourier Transform, you will be informed that the
            program cannot plot the data and the program will exit.

  Usage: plotBeamNoise, <rif>, [ ROF=rof] , ORED=<ored>, [ DR=dr] ,
                        [ DFT=dft] , [ MNDB=mndb] , [ MXDB = mxdb] ,
                        [ PS=psfile] , [ /H]
  where:
   rif       -(string) mandatory: the name of the main RANDI input file
   rof       -(string) optional: main RANDI output filename
              [ default: the file "main.out"]
   ored      -(integer) optional: the code for the omni noise reduction to
              perform on the data before plotting; allowed values are:
                1 - no reduction, plot the data as read
                2 - subtract the coherent omni noise before plotting
                3 - subtract the incoherent omni noise before plotting
              [ default: 1]
   dr        -(integer) optional: the dead reckoning for which data are to be
              plotted; allowed values are:
                <= 0 - the program will count the number of dead reckons in the
                       output file and ask the user for the one to plot
                 > 0 - plot that dead reckoning
              [ default: 1]
   dft       -(integer) variable: mandatory for DFT beamforming when high
              resolution is not used and the number of ships is not 0.  In this
              case if "dft" is not provided it will be asked for; it may be:
```

```
                1 - Coherent results
                2 - Incoherent results
               If this parameter is not required and is missing it will not be
               asked for, and it will be ignored if provided.
               [default: none, the user must select a valid value]
    mndb      -(integer) optional: dB level at the centre of the noise rose;
               if both this and "mxdb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [default: data minimum rounded down to the nearest 10]
    mxdb      -(integer) optional: dB level at the edge of the noise rose;
               if both this and "mndb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [default: data maximum rounded up to the nearest 10]
    psfile    -(string) optional: the name of the output PostScript file
               [default: the output graphics are sent to the screen]
    /H        -(switch) optional: a switch that tells the program to print out
               this "Help" information and exit.  This switch takes precedence
               over all others.
               [default: no "help" information is written and the program runs
               normally]

!ENDHELP

;*******************************************************************************
;
;  "plotShipNoise"

!NEWHELP plotShipNoise

  plotShipNoise                                                - PLOT SHIP NOISE

  This routine plots the shipping noise field from the main RANDI output file.

  Usage: plotShipNoise, <rif>, [ ROF=rof] , [ DR=dr] , [ SNPLT=snplt] ,
                        [ ORED=ored] , [ MNDB=mndb] , [ MXDB = mxdb] ,
                        [ PS=psfile] , [ /H]
  where:
    rif       -(string) mandatory: the name of the main RANDI input file
    rof       -(string) optional: main RANDI output filename
               [default: the file "main.out"]
    ored      -(integer) optional: the code for the omni noise reduction to
               perform on the data before plotting; allowed values are:
                1 - no reduction, plot the data as read
                2 - subtract the coherent omni noise before plotting
                3 - subtract the incoherent omni noise before plotting
               [default: 1]
    dr        -(integer) optional: the dead reckoning for which data are to be
               plotted; allowed values are:
                <= 0 - the program will count the number of dead reckons in the
                       output file and ask the user for the one to plot
                 > 0 - plot that dead reckoning
               [default: 1]
    snplt     -(integer) optional: the type of ship noise plot desired:
```

```
                1 - spike plot
                2 - spike data plotted as a curve  ***NOT YET AVAILABLE***
                3 - smoothed noise data
               [ default: 1]
    mndb       -(integer) optional: dB level at the centre of the noise rose;
               if both this and "mxdb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [ default: data minimum rounded down to the nearest 10]
    mxdb       -(integer) optional: dB level at the edge of the noise rose;
               if both this and "mndb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [ default: data maximum rounded up to the nearest 10]
    psfile     -(string) optional: the name of the output PostScript file
               [ default: the output graphics are sent to the screen]
    /H         -(switch) optional: a switch that tells the program to print out
               this "Help" information and exit.  This switch takes precedence
               over all others.
               [ default: no "help" information is written and the program runs
               normally]

!ENDHELP

;*******************************************************************************
;
;  "plotHzDir"

!NEWHELP plotHzDir

  plotHzDir                                    - PLOT HoriZontal DIRectionality

  This routine plots mean horizontal directionality.

  Usage:  plotHzDir, <rif>, [ ROF=rof] , [ ORED=ored] , [ MNDB=mndb] ,
                  [ MXDB = mxdb] , [ PS=psfile] , [ /H]
  where:
    rif       -(string) mandatory: the name of the main RANDI input file
    rof       -(string) optional: main RANDI output filename
               [ default: the file "main.out"]
    ored      -(integer) optional: the code for the omni noise reduction to
               perform on the data before plotting; allowed values are:
                1 - no reduction, plot the data as read
                2 - subtract the mean omni noise before plotting
               [ default: 1]
    mndb      -(integer) optional: dB level at the centre of the noise rose;
               if both this and "mxdb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [ default: data minimum rounded down to the nearest 10]
    mxdb      -(integer) optional: dB level at the edge of the noise rose;
               if both this and "mndb" are provided and they are equal, the
               program will present the user with the data minimum and maximum
               and force the user to choose values for each
               [ default: data maximum rounded up to the nearest 10]
```

```
   psfile   -(string) optional: the name of the output PostScript file
            [default: the output graphics are sent to the screen]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [default: no "help" information is written and the program runs
            normally]

!ENDHELP

;*******************************************************************************
;
;  "plotTLSL"

!NEWHELP plotTLSL

  plotTLSL                                  - PLOT Transmission Loss/Signal Level

  This routine plots a transmission loss or signal level curve for one radial.

  Usage: plotTLSL, <rif>, [ TLFILE=tlName] , [ MNRN=mnrn] , [ MXRN=mxrn] ,
                  [ MNTL=mntl] , [ MXTL=mxtl] , [ PS=psfile] , [ /H]
  where:
   rif      -(string) mandatory: the name of the main RANDI input file
   tlName   -(string) optional: the name of the file with the transmission
            loss data
            [default: one of the two following files is read,
                "petl.out" - if "mainInput,PEmodel" equals 1
                "fetl.out" - if "mainInput,PEmodel" is not 1]
   mnrn     -(real) optional: minimum range limit to plot; if both this and
            "mxrn" are provided and they are equal, the program will present
            the user with the range minimum and maximum and force the user to
            choose values for each
            [default: 0.]
   mxrn     -(real) optional: maximum range limit to plot; if both this and
            "mnrn" are provided and they are equal, the program will present
            the user with the range minimum and maximum and force the user to
            choose values for each
            [default: maximum range value]
   mntl     -(real) optional: minimum transmission loss/signal level to plot;
            if both this and "mxtl" are provided and they are equal, the
            program will present the user with the data minimum and maximum
            and force the user to choose values for each
            [default: transmission loss/signal level minimum]
   mxtl     -(real) optional: maximum transmission/signal level loss to plot;
            if both this and "mntl" are provided and they are equal, the
            program will present the user with the data minimum and maximum
            and force the user to choose values for each
            [default: transmission loss/signal level maximum]
   psfile   -(string) optional: the name of the output PostScript file
            [default: the output graphics are sent to the screen]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [default: no "help" information is written and the program runs
```

```
            normally]

!ENDHELP

;*******************************************************************************
;
;  "plotDir3D" plots the ambient noise directionality by elevation and azimuth
;              as line plots, and by both as a 3-D plot

!NEWHELP plotDir3D

  plotDir3D                          - PLOT ambient noise DIRectionality (3-D)

  This plots the ambient noise data as a 3-D plot on azimuth vs. elevation
  axes and as two 2-D plots: azimuth vs. intensity and intensity vs. elevation

  Usage:  plotDir3D, <rif>, [ VFILE=vfile] , [ PLOTYP=plotyp] , [ MNHOR=mnHor] ,
                     [ MXHOR=mxHor] , [ MN3D=mn3d] , [ MX3D=mx3d] , [ PS=psfile] ,
                     [ /BW] , [ /H]
  where:
   rif       -(string) mandatory: the name of the main RANDI input file
   vfile     -(string) optional: Vertical directionality data filename: the
              name of the file with data to be plotted, entered either as a
              quote-delimited filename or as a variable containing the file's
              name
              [ default: the file "vert_dir.out"]
   plotyp    -(integer) type of desired plot:
                1 - plot all beams
                2 - plot only real beams
              [ default: 1]
   mnHor     -(real) optional: minimum Y axis value on the Horizontal
              Directionality plot; if both this and "mxHor" are provided and
              they are equal, the program will present the user with the Y axis
              minimum and maximum and force the user to choose values for each
              [ default: data minimum]
   mxHor     -(real) optional: maximum Y axis value on the Horizontal
              Directionality plot; if both this and "mnHor" are provided and
              they are equal, the program will present the user with the Y axis
              minimum and maximum and force the user to choose values for each
              [ default: data maximum]
   mn3d      -(real) optional: minimum noise value to be plotted on the 3-D
              plot - all values less than this value are set to it; if both
              this and "mx3d" are provided and they are equal, the program will
              present the user with the noise minimum and maximum and force the
              user to choose values for each
              [ default: data minimum]
   mx3d      -(real) optional: maximum noise value to be plotted on the 3-D
              plot - all values greater than this value are set to it; if both
              this and "mn3d" are provided and they are equal, the program will
              present the user with the noise minimum and maximum and force the
              user to choose values for each
              [ default: data maximum]
   psfile    -(string) optional: the name of the output PostScript file
              [ default: the output graphics are sent to the screen]
   /BW       -(switch) optional: Black and White option that tells the program
```

```
            to produce a black and white plot
            [default: colour plot]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [default: no "help" information is written and the program runs
            normally]

!ENDHELP

;*******************************************************************************
;
;  "plotAACDF"

!NEWHELP plotAACDF

  plotAACDF      - PLOT Azimuthal Anisotropic Cumulative Distribution Function

  Usage:  plotAACDF, [ ROF=rof] , [ DR=dr] , [ NCUR=ncur] , [ MNAZ=mnaz] , [ MXAZ=mxaz] ,
                     [ PS=psfile] , [ /H]
;;                   [ YMIN=ymin] , [ YMAX=ymax] , [ PS=psfile] , [ /H]
;; see comment out stuff below; same applies here
  where:
   rof      -(string) optional: the main RANDI output filename
            [default: the file "main.out"]
   dr       -(integer) optional: the dead reckoning for which data are to be
            plotted; allowed values are:
              <= 0 - the program will count the number of dead reckons in the
                     output file and ask the user for the one to plot
               > 0 - plot that dead reckoning
            [default: 1]
   ncur     -(integer) optional: the number of curves to plot, spread evenly
            from "mnaz" to "mxaz"
            [default: 5]
   mnaz     -(real) optional: minimum azimuth value to be plotted, which
            must be between the actual data min and max and less than "mxaz"
            [default: data minimum]
   mxaz     -(real) optional: maximum azimuth value to be plotted, which
            must be between the actual data min and max and greater than
            "mnaz"
            [default: data maximum]
;; ymin     -(real) optional: Y axis minimum, which must be zero or greater
;;          and less than "ymax"
;;          [default: 0.]
;; ymax     -(real) optional: Y axis maximum, which must be greater than "ymin"
;;          [default: 10.]
;; The above lines were commented out due to occurrences in "plotAACDF.pro" -
;; see that file for the full, grim details.
   psfile   -(string) optional: the name of the output PostScript file
            [default: the output graphics are sent to the screen]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [default: no "help" information is written and the program runs
            normally]
```

```
!ENDHELP

;*******************************************************************************
;
;  "plotAmNoise"

!NEWHELP plotAmNoise

  plotAmNoise                          - PLOT AMbient NOISE azimuth vs. depth data

  Usage:  plotAmNoise, <rif>, <wl>, [ AZN=azn] , [ WC=wc] , [ WSC=wsc] ,
                       [ REDH=redh] , [ REDV=redv] , [ RDEP=rdep] ,
                       [ PD1=pd1] , [ PD2=pd2] , [ PD3=pd3] ,
                       [ PS=psfile] , [ /BW] , [ /H]
  where:
   rif       -(string) mandatory: RANDI input file
   wl        -(real) mandatory: wind level (dB), whose actual meaning depends on
              "wc"; if "wc" equals:
                1 - "wl" is a wind source level that will be depth corrected
                2 - "wl" is the constant wind omni level for all depths
              If "wl" is not passed in it will be asked for.
   azn       -(integer) optional: the flag used to identify the azimuth file
              being plotted, i.e. "azim_vs_dep_<azn>.dat"
              [ default: 0]
   wc        -(integer) optional: Wind Change choice, with the legal values:
                1 - wind intensity varies with depth, for vertical arrays only
                2 - constant wind intensity for all depths, for all array
                    orientations
              if the array is not vertical the program sets "wc" to 2, and if
              the user had set "wc" to 1 a message is written about the change
   wsc       -(string) optional: wind speed correction filename
              [ default: "windcorr.dat"]
   redh      -(integer) optional: the code for the horizontal polar plot omni
              noise reduction to perform on the data before plotting; allowed
              values are:
                1 - no reduction, plot the data as read
                2 - subtract the omni value from depth "rdep" before plotting
              [ default: 1]
   redv      -(integer) optional: the code for the vertical directionality omni
              noise reduction to perform on the data before plotting; allowed
              values are:
                1 - no reduction, plot the data as read
                2 - subtract the omni value from depth "rdep" before plotting
              [ default: 1]
   rdep      -(real) varies: the depth whose omni value is used to normalize
              the horizontal and/or vertical data plots.
               - If "redh" and/or "redv" is 2 then this is required, if it is
                 missing the program will insist on getting a value from 0 to
                 the maximum depth.
               - If "redh" and "redv" are both 1 then this is not required; if
                 it is provided it will be ignored.
   pd1       -(real) optional: the depth in metres of the first (leftmost)
              horizontal directionality polar plot; if "pd1" is missing or is
              outside the limits of 0. to the maximum depth, the first polar
```

```
            plot is not drawn
            [ default: the first polar plot is not drawn]
   pd2      -(real) optional: the depth in metres of the second (middle)
            horizontal directionality polar plot; if "pd2" is missing or is
            outside the limits of 0. to the maximum depth, the second polar
            plot is not drawn
            [ default: the second polar plot is not drawn]
   pd3      -(real) optional: the depth in metres of the third (rightmost)
            horizontal directionality polar plot; if "pd3" is missing or is
            outside the limits of 0. to the maximum depth, the third polar
            plot is not drawn
            [ default: the third polar plot is not drawn]
   psfile   -(string) optional: the name of the output PostScript file
            [ default: the output graphics are sent to the screen]
   /BW      -(switch) optional: Black and White option that tells the program
            to produce a black and white plot
            [ default: colour plot]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [ default: no "help" information is written and the program runs
            normally]

!ENDHELP

;*******************************************************************************
;
;  "plotOmniSN"

!NEWHELP plotOmniSN

  plotOmniSN                                    - PLOT OMNI Shipping Noise level

  Usage:  plotOmniSN, <rif>, [ OMNIF=omnif] , [ CTYP=ctyp] , [ DR=dr] ,
                      [ OMIN=omin] , [ OMAX=omax] , [ PS=psfile] , [ /H]
  where:
   rif      -(string) mandatory: RANDI input filename
   omnif    -(string) optional: OMNI filename
            [ default: the file "omni.dat"]
   ctyp     -(integer) optional: coherence type to plot; valid values are:
              1 - plot coherent data
              2 - plot incoherent data
            [ default: 1]
   dr       -(integer) optional: the dead reckoning to be plotted.  Valid
            values are:
             <=0 - the program will count the number of reckonings and ask the
                   user for the one to plot
              >0 - plot the specified reckoning
            [ default: 1]
   omin     -(real) optional: minimum omni level to plot; if both this and
            "omax" are provided and they are equal, the program will present
            the user with the data minimum and maximum and force the user to
            choose values for each
            [ default: omni level minimum]
   omax     -(real) optional: maximum omni level to plot; if both this and
```

```
            "omin" are provided and they are equal, the program will present
            the user with the data minimum and maximum and force the user to
            choose values for each
            [default: omni level maximum]
   psfile   -(string) optional: the name of the output PostScript file
            [default: the output graphics are sent to the screen]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [default: no "help" information is written and the program runs
            normally]

!ENDHELP

;*******************************************************************************
;
;  "plot3DTLPL"

!NEWHELP plot3DTLPL

  plot3DTLPL                          - PLOT 3-D Transmission Loss/Pressure Level

  Usage:  plot3DTLPL, <rif>, [FTG=ftg], [DMAX=dmax], [MN3D=mn3d], [MX3D=mx3d],
                  [/NOPOS], [PS=psfile], [/BW], [/H]
  where:
   rif      -(string) mandatory: the name of the main RANDI input file
   ftg      -(integer) optional: field type generator; valid values are:
              1 - FEPE, file "fepe.out" is read and plotted
              2 - SSPE, file "sspe.out" is read and plotted
              3 - FEPE vectorized, file "fepe.out" is read and plotted
            [default: 1]
   dmax     -(real) optional: depth maximum for plotting; valid values are:
              >0 - plot the data up to that depth in km
               0 - use the maximum data depth
              <0 - the program tells the user what the maximum depth is and
                   insists on being provided with a plotting value
            [default: maximum data depth]
   mn3d     -(real) optional: minimum noise value to be plotted on the 3-D
            plot - all values less than this value are set to it; if both
            this and "mx3d" are provided and they are equal, the program will
            present the user with the noise minimum and maximum and force the
            user to choose values for each
            [default: data minimum]
   mx3d     -(real) optional: maximum noise value to be plotted on the 3-D
            plot - all values greater than this value are set to it; if both
            this and "mn3d" are provided and they are equal, the program will
            present the user with the noise minimum and maximum and force the
            user to choose values for each
            [default: data maximum]
   /NOPOS   -(switch) optional: tells the program not to write the source and
            receiver's positions on the page
            [default: the positions are written out]
   psfile   -(string) optional: the name of the output PostScript file
            [default: the output graphics are sent to the screen]
   /BW      -(switch) optional: Black and White option that tells the program
```

```
            to produce a black and white plot
            [ default: colour plot]
   /H       -(switch) optional: a switch that tells the program to print out
            this "Help" information and exit.  This switch takes precedence
            over all others.
            [ default: no "help" information is written and the program runs
            normally]

!ENDHELP
```

# RANDI IDL Graphics Development

## Final Report
## August 2000

**Prepared for:**

Francine Desharnais
DREA
9 Grove St.
Dartmouth, N.S.
B2Y 3Z7

**Prepared by:**

xwave
5475 Spring Garden Road
Suite 601
Halifax, Nova Scotia B3J 3T2

## ABSTRACT

The DREA model RANDI (v3.1) produces as output a number of ASCII and binary files, depending on the manner in which its various parts are run. On its original platform, a Sun SPARC Workstation, the model was supported by a suite of FORTRAN / Sun GKS programs that produced graphics output from the model's output data files. Although RANDI has been converted to run on an HP workstation, the graphics programs cannot be as simply converted because of their reliance on the machine-specific Sun GKS library. However, since the target computer has IDL installed, it was decided to rewrite the graphics programs in IDL. The contract that allowed RANDI to be converted to run on the HP also allowed a start to be made on rewriting the graphics programs in IDL. The current contract allowed for an "upgrade" of the existing IDL routines and the writing of some new ones.

## RÉSUMÉ

Selon la façon dont on utilise ses diverses composantes, le modèle RANDI (v3.1) présente ses données de sortie dans un certain nombre de fichiers ASCII et binaires. Sur sa plate-forme d'origine, soit un poste de travail Sun SPARC, le modèle était soutenu par une série de programmes FORTRAN / Sun GKS qui produisaient des sorties graphiques à partir des fichiers de données de sortie du modèle. Le modèle RANDI a été modifié pour tourner sur un poste de travail HP, mais les programmes graphiques ne peuvent pas être modifiés aussi simplement, car ils dépendent de la bibliothèque Sun GKS propre à l'ordinateur. Toutefois, comme le langage de description IDL est installé sur l'ordinateur HP, on a décidé de réécrire les programmes graphiques en IDL. Le contrat de modification de RANDI pour le faire exécuter sur l'ordinateur HP a aussi permis de commencer à réécrire les programmes graphiques en IDL. Le contrat actuel prévoyait une « mise à niveau » des routines IDL existantes et l'écriture de nouvelles routines.

Sun® is a registered trademark of Sun Microsystems, Incorporated
SPARC™ is a trademark of Sun Microsystems, Incorporated
IDL® is a registered trademark of Research Systems, Inc.
Macintosh™ is a trademark licensed to Apple Computer, Incorporated
PostScript® is a registered trademark of Adobe Systems, Incorporated
VMS™ is a trademark of Digital Equipment Corporation
Windows® is a registered trademark of Microsoft Corporation

## EXECUTIVE SUMMARY

RANDI 3.1 is an underwater ambient noise model. It can be used to predict acoustic noise levels and 3-D directionality for shallow and deep water environments, and to predict the low to mid frequency response of sonar receivers to that noise field.

RANDI 3.1 was originally written to run on a Sun SPARC Workstation supported with Sun GKS graphics library. The model had been converted for an HP environment to be used more readily at DREA for sonar research. Under this contract, the RANDI graphics routines were rewritten in IDL to exclude the dependence of the original programs to the Sun GKS library.

As a result, DREA has a functional copy of RANDI ready to be used to model 3-D ambient noise fields for sonar research..

## SOMMAIRE

RANDI 3.1 est un modèle de bruit ambiant sous-marin. Il peut être utilisé pour prédire les niveaux de bruit acoustique et leur directionalité en trois dimensions pour des environnements d'eau profonde et peu profonde, et pour prédire la réponse acoustique de récepteurs sonar à ce champ de bruit aux fréquences basses et moyennes.

Sur sa plate-forme d'origine, soit un poste de travail Sun SPARC, le modèle était soutenu par une bibliothèque de graphiques Sun GKS. Le modèle a été modifié pour tourner sur un poste de travail HP, pour facile à utilisé pour la recherche sonar au CRDA. Sous le contrat actuel, les programmes graphiques ont été réécrit en IDL pour exclure la dépendance des logiciels originels à la bibliothèque Sun GKS.

Le CRDA a maintenant une copie fonctionnelle de RANDI prête à être utilisée pour modeliser les champs de bruit ambiant en trois dimensions pour la recherche sonar.

## Table of Contents

1.0 INTRODUCTION ..... 1
2.0 PRODUCING GRAPHICS ..... 3
2.1 Running **plotRANDI** ..... 3
2.2 General Rules Related to **plotRANDI** ..... 4
3.0 MODIFICATIONS TO INHERITED **plotRANDI.pro** ..... 6
4.0 **plotRANDI** EXPANSION ..... 9
5.0 **plotRANDI** CONCERNS ..... 10
5.1 **plotShipNoise** ..... 10
5.2 **plotAACDF** ..... 10
5.3 **plot3DTLPL** ..... 11
6.0 **plotRANDI** - FORTRAN/GKS CODE COMPARISONS ..... 12
7.0 SUGGESTIONS FOR FURTHER WORK ..... 14
8.0 REFERENCES ..... 16

APPENDIX A - SETUP REQUIREMENTS ..... 17
APPENDIX B - HELP INFORMATION ..... 19

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

## DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e g Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8 )

Xwave, 5475 Spring Garden Road, Suite 601, Halifax, NS

2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable)

UNCLASSIFIED

3 TITLE (the complete document title as indicated on the title page Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title)

RANDI IDL Graphics Development

4. AUTHORS (Last name, first name, middle initial If military, show rank, e g Doe, Maj John E )

Colin Calnan

| 5. DATE OF PUBLICATION (month and year of publication of document) | 6a. NO OF PAGES (total containing information Include Annexes, Appendices, etc) | 6b NO OF REFS (total cited in document) |
|---|---|---|
| December 2000 | 29 (approx.) | 1 |

7 DESCRIPTIVE NOTES (the category of the document, e g technical report, technical note or memorandum If appropriate, enter the type of report, e g interim, progress, summary, annual or final Give the inclusive dates when a specific reporting period is covered)

CONTRACTOR REPORT

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development Include address)

Defence Research Establishment Atlantic
PO Box 1012, Dartmouth, NS, Canada B2Y 3Z7

| 9a PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written Please specify whether project or grant) | 9b CONTRACT NO (if appropriate, the applicable number under which the document was written) |
|---|---|
| 1ca11 | W7707-007913/001/HAL |

| 10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity This number must be unique to this document ) | 10b OTHER DOCUMENT NOs (Any other numbers which may be assigned this document either by the originator or by the sponsor ) |
|---|---|
| DREA CR 2000-160 | |

11 DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

( X ) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document This will normally correspond to the Document Availability (11) However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected)

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

The model RANDI (v3.1) produces as output a number of ASCII and binary files, depending on the manner in which its various parts are run. On its original platform, a Sun SPARC Workstation, the model was supported by a suite of FORTRAN / Sun GKS programs that produced graphics output from the model's output data files. Although RANDI has been converted to run on an HP workstation, the graphics programs cannot be as simply converted because of their reliance on the machine-specific Sun GKS library. However, since the target computer has IDL installed, it was decided to rewrite the graphics programs in IDL. The contract that allowed RANDI to be converted to run on the HP also allowed a start to be made on rewriting the graphics programs in IDL. The current contract allowed for an "upgrade" of the existing IDL routines and the writing of some new ones.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

RANDI
IDL graphics
Ambient noise
Underwater noise

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM